Joint model-driven design and real experiment-based validation for a secure UAV ad hoc network routing protocol

Jean-Aimé Maxa, Mohamed-Slim Ben Mahmoud, Nicolas Larrieu

JOINT MODEL-DRIVEN DESIGN AND REAL EXPERIMENT-BASED VALIDATION FOR A SECURE UAV AD HOC NETWORK ROUTING PROTOCOL

Jean-Aimé Maxa[1] [2] *, Mohamed Slim Ben Mahmoud*[1] *and Nicolas Larrieu*[1]
[1] *ENAC, TELECOM/Resco, F-31055 Toulouse, France*
[2] *Univ de Toulouse, F-31400 Toulouse, France*

Abstract

UAV Ad hoc Network (UAANET) is a wireless ad hoc network composed of Unmanned Aerial Vehicles (UAVs) and Ground Control Station (GCS). Compared to the standard Mobile Ad hoc NETworks (MANETs), the UAANET architecture has some specific features that brings exciting challenges to communication architecture design. One of them is the design challenge of a UAANET routing protocols. It must find an accurate and reliable route between nodes in a timely manner to exchange data traffics. It must also be secured to preserve efficiency in the presence of malicious attackers and provides data integrity and authentication.

Furthermore, UAANETs must be certified in the near future to act as autonomous systems without a dedicated safety pilot and to be authorized to fly in the national airspace. In such a context, in this paper, we contribute to the certification of the secure UAANET communication system software using a Model-Driven Development (MDD) approach and real experiments based validation. The validation process followed uses sequentially formal verification methods and real-world experimental results. The objective is to evaluate the routing protocol efficiency to a set of unexpected hazardous issues that come with the real environment.

Introduction

UAVs investigations began after the first world war with preliminary prototypes. Since then, technological and research advances in embedded systems help to produce small UAVs with highly effective capacities. A small UAV is a pilot-less aerial vehicle that has a set of micro electromechanical systems. It can be controlled either autonomously by on-board computers or remotely controlled by a distant operator. When several UAVs are communicating with each other via wireless links, they dynamically form a temporary multi-hop radio network called UAV Ad hoc NETwork (UAANET). Compared to the standard Mobile Ad hoc Network (MANETs), UAANETs have some specific features (3D mobility model, low node density, intermittent network connectivity) that brings interesting challenges to the communication architecture design. Indeed, in order to achieve UAANET missions (for instance surveillance and reconnaissance [1]), UAVs must relay network and payload packets between them to the Ground Control Station(GCS). Since the existing routing protocols designed for MANETs fail in tracking network topology changes [2], adapted routing protocols are required to establish an accurate and reliable route between nodes.

Furthermore, the routing protocol also needs to be secured as it relay critical data traffics (command/control (c2) traffics, payload [1] traffic and the routing protocol traffic). This is because, in a wireless ad-hoc environment in which there is no fixed infrastructure, the wireless links are prone to link attacks, which consist of passive eavesdropping and active interfering. As a result, routing control packet needs to be authenticated to verify both the identity of the message originator and the message integrity to avoid data tampering and modification.

Likewise, UAANETs applications offers a large possible civil and public domain applications in which one or several UAVs may be deployed [3]. One of the obstacle that stands for its deployments lies on certification restrictions. Indeed, UAANETs must be certified to act as autonomous systems without a

[1] In order to avoid misunderstanding, note that the payload from an UAS point of view is quite different from the network payload traffic, which is the amount of additional data required by traffic control.

dedicated safety pilot or to be simply authorized to fly in the general airspace. On a regional French scale, such a system needs to be fully compliant with the DGAC[2] regulations.

In such a context, the UAANET communication system must satisfy both UAANET network (routing and security) and validation requirements. Accordingly, in this paper, we provide a case study of designing a secure UAV Ad hoc routing protocol applying Model Driven Development (MDD). Our objective is twofold: on the one hand, to minimize certification and evaluation efforts by providing a formal verification capability which ensures quality and conformance of the routing protocol, and on the other hand, to validate the efficiency of the routing protocol through real UAANET test experiments. This validation process is complementary of the formal validation step in order to assess how our software solution behave in real outdoor experiments. It should be noted that this paper is part of the Secure UAV Ad hoc Network (SUANET) project presented in [4].

The paper is organized as follows. In section 2, we describe the UAANET communication architecture, routing and security requirements. Section 3 highlights the current UAS certification state of the art. The model driven design of our secure UAANET routing protocol will be detailed in section 4. Then, we will describe in section 5 and section 6, the real-word experiments validation and performance evaluations of our routing protocol. Finally, in Section 7, we provide our conclusions and an overview of our future research perspectives.

UAANET communication architecture

In [5], survey of communication networks of small UAVs are presented. The authors characterize different network architectures: direct communication, cellular and ad hoc networking. Direct communication scheme and cellular networks have several flaws. They require a certain amount of bandwidth for each UAV to support a high node density in the network. Specifically in the direct-link architecture, UAV-to-UAV communications are not possible as data traffic must be routed to the GCS. In regards to the cellular architecture, the existing mobile phone infrastructure is not intended for air-to-ground communication. Since a single UAS transmitter can sweep a wide area with its signal, a UAS deployments may need a dedicated cellular infrastructure. But, the expensive implementation of these base stations is a financial handicap.

In order to address the weaknesses of the communication architectures discussed above, a UAANET can be used for UAV swarm (depicted in Figure 1) in which UAVs and GCS are communicating between each other without the need for a fixed infrastructure. Compared to other communication networks cited above, UAANETs have several advantages. The scalability is ensured thanks to the fast mobility of UAVs to cover a vast area rapidly. Also, the reliability is improved because the failure of one UAV does not affect the whole network. As regards the bandwidth, it can be reused more often and thus more efficiently due to the multi-hop communication. From a security viewpoint, the absence of a central node within the network decreases the risk of attack on a single point of failure. Each end system (UAVS and the GCS) is responsible for the network integrity and authentication.

These features make UAANET the most appropriate solution for UAVs communication. However, it raises a challenging networking problem.

Routing Protocol Challenges in UAANET

Several dynamic routing mechanisms are available for UAANETs [6]. Topology-based mechanisms, such as proactive, reactive and hybrid routing are the basis of numerous protocols. Nonetheless geographical routing could also be efficient in specific contexts.

In [7], we introduced an emulation-based performance evaluation of MANETs routing protocols for UAANETs. This realistic study considers the Linux kernel networking stack requirements, the protocol implementations, background traffics, real time execution features and a realistic mobility model. The experimental results showed that AODV [8] routing protocol suits better in UAANET compared to OLSR [9] and DSR [10]. As AODV being the protocol the most reactive to topology changes and having a limited overhead, we concluded that AODV was the most suitable routing protocol for our scenario. As several studies proved that AODV can be outperformed by proactive protocols with a large num-

[2]DGAC: "Direction Générale de l'Aviation Civile" which is equivalent to the American FAA (Federal Aviation Administration)

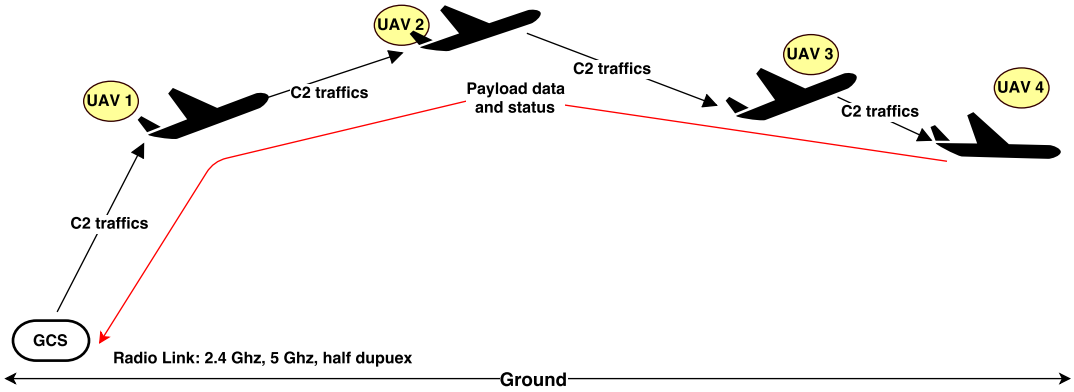

Figure 1. UAANET Architecture

ber of nodes, its on-demand mechanism allows a faster response during UAANETs disconnection. Our mobility pattern and our emulation system certainly affected the measures, but we found similar results to those exposed in [11]. Accordingly, in [12], we introduce the model design of SUAP (Secure Uav Ad hoc routing Protocol) which is a secure UAANET routing protocol based on AODV.

Security Challenges in UAANET

The ultimate objective of a routing protocol is to find an accurate and reliable route while assuming that nodes in the network are friendly and cooperative. However, such assumption might not always be true as UAANET use an unprotected wireless medium and lack of fixed infrastructure to separate inside nodes from the outside (the attackers). Also, because of the critical level of c2 traffic exchanged, its security must be insured at all cost. Note that there can be different motivations for attackers to breach UAANET. For instance, an attacker may attempt to take control of UAVs by capturing the control and command traffics. This can be achieved through eavesdropping and packet forwarding attacks [13].

Generally, securing UAANET is a challenging task due to its cooperativeness characteristic and its uncontrollable environment. Unlike wired networks where attackers need to gain access to the physical medium to execute any types of attacks, in UAANET, an intruder can easily eavesdrop the exchanged traffic through the unprotected wireless links. From our best knowledge, only few secure UAANET routing protocol [4] has been proposed. It can be either a new routing protocol that is designed from scratch with the security as one of its goals or a secure extension of an existing routing protocol. The latter approach is the most used one as it uses a routing protocol like AODV, DSR and OLSR, considered by the IETF MANET group for standardization. Thus, securing those routing protocols requires assessing attacks specific to that protocol and securing them accordingly. This latter approach is used in this paper to create the SUAP routing protocol.

UAS Certification Challenges

To apply for a certificate, UAS is divided into several parts ranging from electronic components, hardware components to software components that need to possess an airworthiness approval [14]. The advantage of evaluating these components separately is two fold: first, it allows the manufacturer of the different UAS components (engines, propellers) to design components that can be used in several UAS designs. Second, it allows the UAS assembler (in our case Delair Tech company [15]) to be involved only with the certification of the system, not with each component itself. This reduces the effort needed by the UAS designer to apply for a certificate. Note that a UAS can only be certified by the aviation authorities of the country where the principal place of business of the designer is located. Accordingly, on a regional French scale, our UAS communication system needs to be validated by the DGAC. Although the French UAV professional civil federation[3] is yet to propose a specific validation and certification standards for UAANETs, the process and safety standards depicted

[3]The French organization responsible for UAS certification within DGAC: http://www.federation-drone.org/ certification

within the DO 178C [16] can be applicable for UAANETs [17]. DO-178 is a commercial aviation standard used to certify aeronautical systems whose focus lies on Model Based Development and verification supplement with formal verification methods.

Note that, Delair Tech company has previously certified its DT-18 UAV to fly out of sight of the GCS (i.e, the operator). However, the UAANET architecture induces a modification of the existing communication system. As a result, a new validation and certification process is required. Consequently, the secure routing protocol that we are currently building must be the subject of strict validation process. We considered such requirements by using a qualified software development processes using model driven development.

Model driven development of secure UAANET routing protocol

The most used current Linux implementation of AODV (AODV-UU) is a source code that has not been carefully validated and certified. Also, there is no guarantee that all the functional requirements specified by the IETF standards have been implemented. Thus, there is no conformance testing between specification requirements and source code. Consequently, its direct deployment into UAANET is not advisable as it might fail during flight operations and cause meaningful damage. Different compliance testing methods have been developed for distributed communication systems [18]. However, most of them are goal-oriented testing techniques which consist of choosing a specific property of the target system that is likely to be faulty. One way to overcome this issue is to use MDD approaches that can automatically generate, deploy and verify code by taking as inputs a high-level model.

MDD claims the use of models as primary artifacts in the development process. A system model composed of block diagrams and state charts is the focus of the development process, from requirement specifications to simulation testing and integration. When used with a qualified code generator, it generates a high level codes which facilitate the early verification of the design through formal verification tools. It also execute model-and-code consistency checking for system verification purposes.

Design methodology

Our design is based on Matlab Simulink and Stateflow frameworks. Simulink is composed of a block-diagram environment which allows us to accurately design the routing protocol while the stateflow frameworks allow us to define a finite number of states in the algorithm which are changed based on a defined condition.

Figure 2 represents the model driven development workflow and schematizes the integration process. It is composed of the seven different steps. In the first step, the specification is validated and partitioned into several subsets of the requirements. Each subset has its own unit test objectives. Then, in the second steps, each partition is designed into a high-level model using graphical descriptive language provided by Simulink and stateflow tools. To meet the SUAP security requirements and routing features, we used both security block architectures depicted in Figure 5 and generic routing protocol architecture as shown in Figure 6. The network layer is divided into several blocks exchanging messages between them and adjacent layers. Each block is designed as a state machine which analyzes incoming requests, then sends them in each iteration. The description of these design architectures is detailed in the next section .

Then, during the third step, each block model is converted into C library source code by Mathworks Embedded coder. Note that these source codes are independent of any operating system or hardware architecture. The next step (called glueing) consists of linking the generated code to the kernel space of our target operating system (Embedded Linux generated from OpenEmbedded). In the fifth step, the library code and glueing code are combined into binary files. The next step consists of aggregating the previous binary files with our target operating system to provide a final binary image. The last step is the execution of binary image into the target hardware for verification and validation.

Secure routing protocol design architecture

There have been several proposals to ensure the security of AODV [19]. They provide message authentication and integrity by means of cryptographic techniques for route discovery and route maintenance packets. However, most of them are vulnerable against wormhole attacks [20]. The wormhole

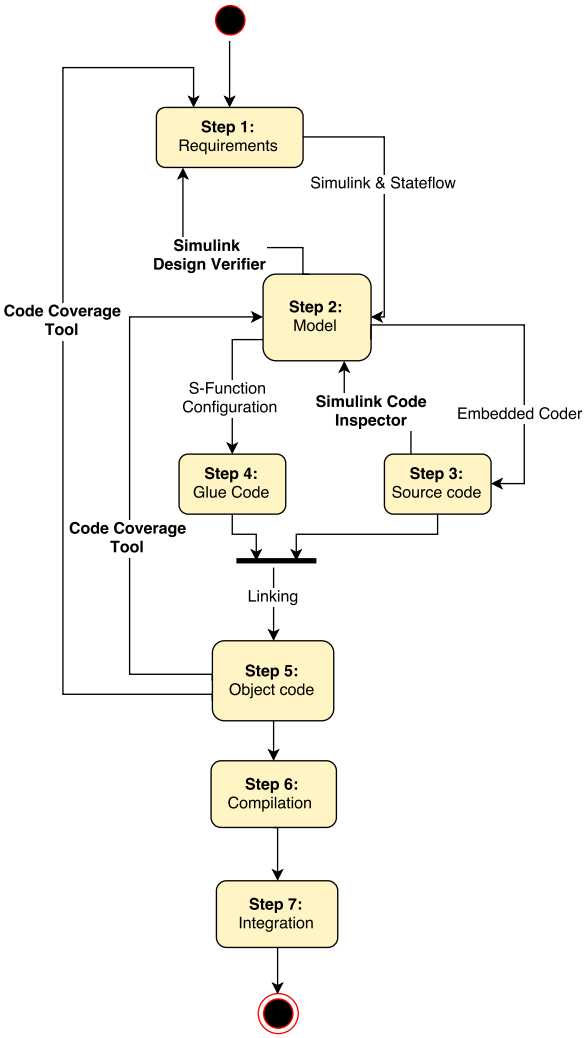

Figure 2. Set of MDD Tools Used to Design This Secure Routing Protocol

attack involves two attackers who perform a colluding attack. One attacker record packets at a particular location and replay them at another attacker by using a high-speed private network. This tunnel between two colluding attackers is referred to as a wormhole. Because of the broadcast nature of the radio channel, attackers are able to receive data packets not addressed to them. Figure 3 demonstrates an example scenario of this attack within UAANETs, where A1 and A2 are the colluding attackers while the GCS is the victim node. Even though, each node encapsulates each packet with cryptographic keys, wormhole attackers can breach UAANET security as there is no key required to process wormhole attacks. It should also be noted that wormhole attack is easy to execute as a commercial high-gain antennas configured in specific frequencies can eavesdrop the frequency and receive the signals send by the GCS and the UAVs.

Furthermore, the following assumption is considered in our network and attacker model:

- UAVs and GCS are coming from the same manufacturer which allows us to put aside the security issue from selfish nodes. Apart from being captured or controlled by an attacker, a node will not change its behavior and will always cooperate with its neighbors;
- Nodes have sufficient energy power and network resources (i.e bandwidth) to perform cryptographic computation;
- All UAVs utilizes omnidirectional antennas. Communication range is r and cannot exceed Dmax (r < Dmax). Dmax is a maximum one-hop range;
- Nodes rely on efficient symmetric and asymmetric cryptography algorithm for encryption/decryption, authentication and hashing. We envision to use RSA algorithm for message authentication and SHA-256 as a hash algorithm.
- All nodes are clocks synchronized. This is possible with the presence of GPS on-board of UAVs and GCS;
- We assume that there is an efficient and reliable key management within the network to share, to manage and to revoke node cryptographic keys;
- Routing control packet confidentiality is not insured. Indeed, routing packets are processed in real time by the flying UAVs. As such, even if an attacker is able to eavesdrop the message, its action is limited in passive mode because in the future, the past information is no longer valuable.
- Hash function H is only known by legitimate nodes and pre-loaded with keys at the bootstrapping;
- In regards to attackers capabilities, we consider the work of Cordasco et al. in [21] based on Dolev-Yao model in which they present a topology and protocol agnostic model that takes into account a real-world scenario. Accordingly, we considered that the following attacks are possible: data traffic disclosure, routing information disclosure, performance degradation and topol-

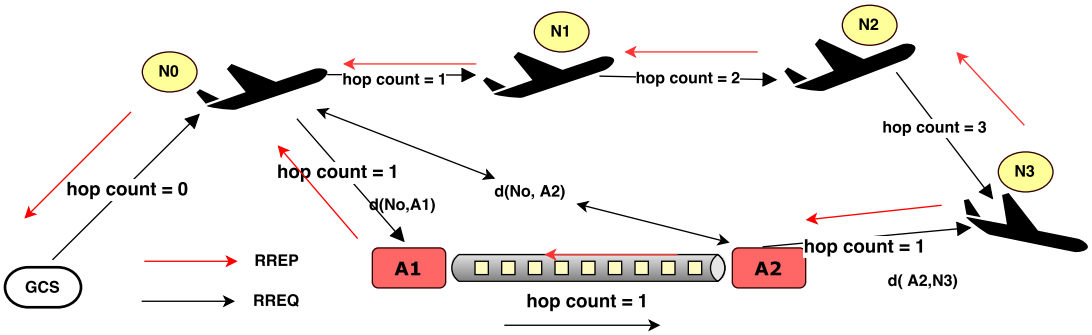

Figure 3. Illustration of Wormhole Attack in UAANETs

ogy modification.

By taking into account, those requirements, we propose the SUAP routing protocol which is a security extension of the AODV protocol, based on public key cryptography, hash chains and geographical leashes [22]. It uses digital signatures for non-mutable fields[4] and hash chains for mutable fields (i.e the hop count). A node that generates a routing message signs it with its private key, and the nodes that receive the message verify the signature using the sender's public key. The hop count cannot be signed by the sender, because it must be increased at every hop. Therefore, a mechanism based on hash chains is used. As such, SUAP is vulnerable against wormhole attacks. Accordingly, a version of geographical leashes is used to estimate the correlation between the travelled distance and the hop count value. In order to do so, SUAP requires each node in the network to be tightly synchronized and maintain a local connectivity with its direct neighbor.

Moreover, two distinct mechanisms are used to protect against wormhole attacks. The first one is used for hello and route error messages. In our network model, we make the assumptions that, at the bootstrapping, nodes directly send hello messages in broadcast to reckon their directors (one hop) neighbors and detect link breaks. To protect these packets from wormhole attacks, we use a mechanism that analyzes the correlation between hop count and distance traveled by the packet. When sending messages, each node includes its actual location information. To protect from malicious modification, message fields are signed (including the geographical position). Figure 4 illustrates the format of modified beacon messages.

[4]Routing message fields that does not change throughout the network. For instance, Destination IP Address, Source IP address, destination sequence number, etc.

Table I. Notation Table

Paremeter	Description
hc	Hop count
Dmax	One hop distance maximum
Rij	Distance between nodes i and j
d(No, A1)	Distance between the first target and the first attacker
d(A1,A2)	Distance between the two attackers
d(No, A2)	Distance between the first target and the second attacker
d(A2,D)	Distance between the second target and the second attacker
Pi	The amount of distance traveled by a packet at the time t
T	The total distance of the legitimate route
Dw	The total length of the path through the wormhole link

To illustrate our proposition, we considered the Figure 3 and the notation in table I. The connectivity between two legitime nodes can be expressed by (1), with $R_{i,j}$ is the current communication range.

$$c(i,j) = \begin{cases} 1 & si \quad R_{i,j} \leq Dmax \\ 0 & si \quad R_{i,j} > Dmax \end{cases} \qquad (1)$$

The presence of wormhole link modify this condition to (2).

$$c(i,j) = \begin{cases} 1 & si \quad R_{i,j} \geq Dmax \\ 1 & si \quad R_{i,j} > Dmax \end{cases} \qquad (2)$$

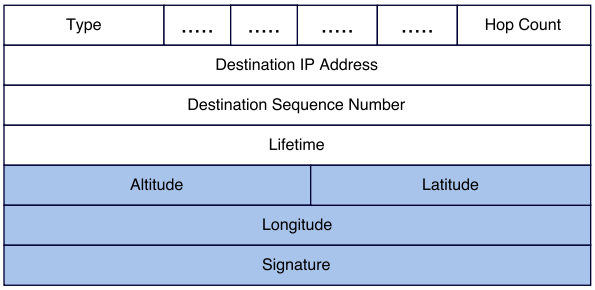

Figure 4. Format of SUAP Modified Hello Message

Furthermore, we have :

$$d(No,A1) \leq Dmax$$
$$d(A2,N3) \leq Dmax$$
$$d(No,A2) > Dmax$$
$$d(A1,N3) > Dmax$$

It results that.

$$d(No,A1)^2 + d(A1,A2)^2 > Dmax^2$$

thus,

$$d(A1,A2) > Dmax - d(No,A1)$$

We have (3) :

$$Dw = d(A1,A2) + d(No,A1) + d(A2,N3)$$

$$Dw > Dmax \quad (3)$$

Knowing that

$$T = \sum_{i=0,j=0}^{n} R_{i,j}$$

- When the node N0 send the packet, $To = Ro1$ that corresponds to hc = x + 1 with $x \in \mathbb{N}$;
- When the node N1 send the packet, $T1 = To + R12$ that corresponds to hc = x + 2;
- When the node N2 send the packet, $T2 = T1 + R23$ that corresponds to hc = x + 3;

thus,

$$\frac{T}{Dmax} - 1 \leq hc < \frac{T}{Dmax} + 1 \quad (4)$$

By taking into account the inequality (3), we can compare the hop count value present in the packet and the hop count value computed on the traveled distance by following the corresponding value depicted in the Table II and the formula (4). If there is a difference, the wormhole link is detected, and the packet is rejected. Otherwise, the link is considered as free of wormhole and the signature verification process begins. Note that because of the position information included in the packet, it is possible to compute the relative distance between 2 neighbor nodes using 3D Euclidean distance.

Table II. Mapping Table Between the Distance Traveled and Hop Count

T	Hop count (hc)
$0 < To \leq Dmax$	0
$Dmax < T1 \leq 2Dmax$	1
.....	.....
$(n-1)Dmax < T_{n-1} \leq (n+1)Dmax$	n-1

Table III. SUAP Request Packet Signature Extension Fields

Field	Value
Type	64
Hash function	hash function selected by the sender node. It is used to compute the hash chain field
Signature	The signature of all the non-mutable fields
Hashnew	Hashnew = H [CurrentNode, NextNode, Hashold] CurrentNode is the address of node sending the request packet. It can be its public key or its IP Address. The Nextnode is the next node public key or IP Address. Hashold is the previous chain element received from the previous node
Hashold	It is the previous chain element received from the previous node. When receiving packets, nodes change the value of Hashnew into Hashold
Hop Count	The actual hop count of the packet. It is the number of times the hash is performed

The second mechanism is used during the route discovery process in which request and response messages are exchanged. In this mechanism, nodes do not need to send its geographical position. Instead, each node assumes that its local connectivity is secure thanks to the neighbor information provided by the previous mechanism. Each node then send in unicast all route discovery packets to its direct neighbors. Each node also includes the address of the next hop to which the message is forwarded and apply a hash chain to the packet mutable fields. The non-mutable fields are signed as stated previously. An illustration of the request message is shown in Table III. The source node appends its own address and the next node address to the hash chain called Hashnew. It also includes the Hashold (which is the previous Hashnew) within the packet. When an intermediate node says Ni receives a request, it checks its signature and verify the Hash chain. It re-computes the Hash chain with H [previousnode, MyIPAddress, Hashold] and verify if it has the same result as the one included in the packet.

Considering the Figure 3, we compute the hash chain as the following:

The GCS node executes the following operation:

- Select a H hash function;
- Compute $Oldhash = H(seed)$, seed is a value selected randomly by the sender;
- Compute Hashnew = H(GCS, N0, Oldhash), N0 is the next node address;
- Compute message S to N0: [64, H, signature, Hashnew, Oldhash]

When the node N0, receives the packet, it processes the following operation:

- Integrity verification by computing Hashverifier = H [previousnode, actualnode, Oldhash] and verify the result compared to Hashnew. Since, we use a one way hash function, the slightest change would lead to difference. Besides, the hash function is only known by legitimate node.
- If Hashverifier = H[GCS, N0,Oldhash] = Hashnew, it indicates that the link is free of wormhole attack. Otherwise, it means that the packet has been transmitted via a wormhole link. The packet is then rejected.
- Assign the new Oldhash = Hashnew
- Compute the new Hashnew with Hashnew = H[N1, N2, Oldhash]

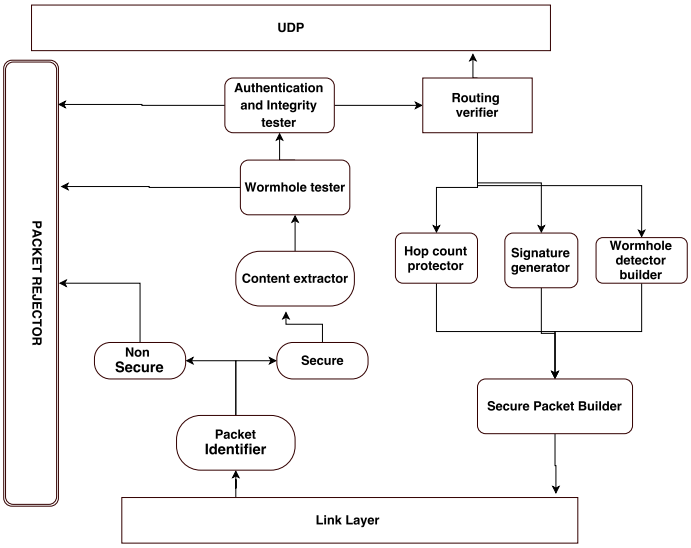

Figure 5. Secure UAANET Routing Protocol Design Architecture

The operation is repeated until the packet reaches the destination. The same mechanism is also used for the response packet. As regards the exact value of the hop count values, it can be inferred from the number of times that the hash was used for verification. It can also include in the hash chain computation. Note that this mechanism can also be efficient against wormhole attacks involving only one attacker [23].

Security block design architecture

In order to specify our specification into high-level models, we use a dedicated security design architecture for the security part (depicted in Figure 5) and a dedicated generic routing architecture for the routing part (shown in 6). The specification follows the IETF draft [8] in which each requirement (messages, tables, parameters, extension specification) has been meticulously respected. Our system includes several blocks transmitting signals between them denoting the security and network requirements specifications. They also contain several instances of stateflow graphic representation. These blocks describe the possible protocol behavior as will detail in the following. To give a general idea of the complexity of the SUAP model specification, we present in the Table IV some significant metrics of the global system.

In the security architecture design, each packet received from the link layer must be verified by a

Table IV. Metrics of the SUAP Protocol Specification

Code lines	6236
Blocks	30
Procedures	188
States	30
Signals	36
Macro definitions	14

module called "Packet identifier" to check whether or not the packet has a security extension. In case, it does not contain a sufficient security extension, the packet is rejected by the Packet rejector module.

The packet rejector module role is to delete suspicious packets. If the packet has its security extension activated, its content is extracted from the Content extractor module. This module explicitly identified which part contains the signature, the hash chains and the location information for the wormhole detector.

Then, the packet is verified by the wormhole tester module which mainly compute either the association between the hop count and the distance traveled by the packet (if beacon and route error messages are exchanged), or compute the actual hash chains for the message. In this step, if the packet fails to prove its required security level against wormhole attacks, it is redirected to the packet rejector module. Otherwise, it is sent to Authentication and Integrity tester module. In this block, the authentication of non-mutable fields and integrity of mutable fields is verified. If all the information within the security extension is valid, the packet is directed to the routing module to check whether or not the packet has reached its final destination.

Regarding routing blocks, The basic routing process is modeled as follows:

- Demux block accepts packet from transport layer (when the actual node needs to search a route towards a given destination) or the link layer. Then, the Demux block verifies the type field of the packets to determine if the packet is for route discovery block or route maintenance block.
- Route discovery and Route maintenance block: Upon the reception of a packet from the "Demux" block, "Route Discovery" and "Route Maintenance" process the same operation. They analyze the encapsulated routing information within the packet to determine its type: RREQ, RREP, HELLO, or RERR packet.
- RREQ Mgmt, Hello Mgmt, RREP Mgmt, RERR Mgmt: These blocks determine if the packet carries sufficient routing information and reached its final destination. Based on this result, the packet is transmitted, to the "Packet send handler."
- Packet send handler: The "Packet send handler" block is used to retrieve the packet and to update its fields according to its destination, whether to the transport or access layer depending on whether or not the current node is the final destination node. In case the packet has reached its final destination, it is transmitted to the application layer through a user-space daemon. Otherwise, it is forwarded to the next node in the routing table.

It is important to underline that some functionalities described in the AODV specification has been omitted (e.g. multicast support, gratuitous RREP messages or packet extension) as they are not necessarily needed in our network model.

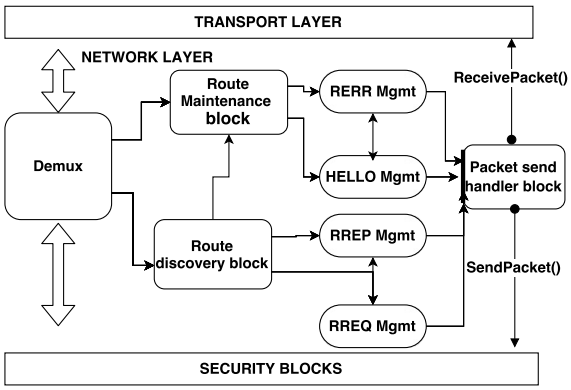

Figure 6. UAANET Routing Protocol Design Architecture

Formal verification process

A key benefit of MDD is early verification. Verification begins when block models are created and simulated using unit tests based on high-level requirements. In our case, this verification is processed by additional tools as shown in Figure 2:

- Simulink Code Inspector: it is processed to perform model-and-code consistency checking. It examines blocks, state diagrams, parameters, and settings in the model to determine whether

they are structurally equivalent to the operations, operators, and data in the generated code. Then it generates a traceability documentation with respect to the DO-178C reference document that can be used for certification purposes.
- Simulink Design Verifier: it uses formal checking tools to identify model design errors. It detects, blocks that lead to errors, such as dead logic, integer and fixed-point overflows, division by zero.
- Model coverage tool: it gives line coverage information. It indicates which part of the model is depicted by a given line of code. This is useful when an error has been identified by the Simulink Design Verifier to locate which part of the model is faulty.

UAANET real-world experiments

Routing Implementation details

To execute the source code obtained through MDD approach, it is necessary to cross-compile and cross link the embedded Linux C code into the target hardware, which is an ARM board[5]. This cross-compilation is executed with OpenEmbedded framework [24] which also generates[6] a lightweight and efficient Linux distribution for the ad hoc network communication. It should be noted that our implementation is mainly based on the Linux Netfilter package through a kernel module. We attach each entry and output of our model to the set of hooks points within the Linux protocol stack. Compared to kernel modification methodologies [25], our implementation does not necessitate unnecessary communication (for instance the exchange of ARP packet) and can be installed and initialized through the user-space daemon as in AODV-UU.

Experimentation details

The UAANET architecture that has been set up is shown in Figure 8. The network is composed of 3 nodes, including 2 fixed wing UAVs (DT 18 [14] developed by the Delair Tech company) and one GCS (also from Delair Tech). The DT-18 is a lightweight UAV capable of long-range flight whose characteristics are detailed in Table V. The first UAV called *Dr1* is flying at a distance of 250-500 meters from the GCS whereas the second UAV *Dr2* flies at a distance 1500-2000 meters from the GCS. Hence, the distance between the two flying UAVs is approximately 1500m. The average altitude of UAVs is estimated at 1127 feet (345 m).

Table V. Main Characteristics of DT-18 UAVs

Characteristic	Value
Model	DT-18
Payload	250 g
Range	100 km
Cruise speed	50 km/h
Wind	up to 45 km/h
real-time payload transmission	up to 15 km. Extension to 100 km
Autopilot	Delair-Tech technology
Onboard computer	payload and communication control, 1 GHz
Experiment duration	30 minutes

Furthermore, regarding the different types of traffics exchanged between nodes, there are 5 types of data traffics (depicted in the table VI) including:

- Heartbeat traffic: It is a 100 Bytes data packet sent both by the GCS and by each UAVs at 1 Hz frequency. Each node keeps sending heartbeat packet to find out whether or not it is connected to its associated neighbor. If a number of heartbeats messages are missing, the UAV failsafe (can be) is triggered to land the UAVs at the launch point.
- Geographical reference: It is a 80 bytes packet for determining UAVs positions. It is sent from the UAVs to the GCS. 3 packets are sent per second
- Command and control traffics sent to each UAV from the GCS. 80 Bytes sent per second. It consists of a flight configuration and waypoint sent by the operator through GCS desktop software.
- Network traffic: It consists of routing control packets exchanged between nodes for route discovery and route maintenance.
- Payload traffic (video traffic) exchanged from Dr2 to the GCS through Dr1. We sent 25 UDP packets (each packet has 1400 Bytes) per second.

On each UAV, we mount an ARM-based computer-on-module produced by Phytec Inc. It runs

[5]This is because, the unit test of the routing protocol has been processed with a X86 architecture. As a result, the cross compilation is necessary to add the missing headers required by the cross compiler and the cross linker

[6]configure and build

Table VI. The Different Flows Exchanged During the Flight Test

Type of traffic	Source —- Destination	Size	Flow rate
Heartbeat or Tick	GCS —- Dr1 GCS —- Dr2	64 Bytes	1 packet/s
Geographical Reference (Georef)	Dr1 —— GCS Dr2 —— GCS	80 Bytes	3 packets/s
C2	GCS—- Dr1 GCS —- Dr2	80 Bytes	1 packet/s
Video	Dr2 ——Dr1—— GCS	1400 Bytes	25 UDP packets/second width=720,height=576
Network	Exchanged between Dr1, Dr2 and the GCS	Request: 66 bytes Response: 62 bytes Hello : 62 bytes Error: 54 bytes	1 packet/s for the hello Request and Response and Error packets are exchanged during disconnection (route loss)

a customized Linux distribution compiled and built with OpenEmbedded framework. We also attached a HD camera and a Wi-Fi radio interface module. The Wi-Fi radio interface being used is the Mikrotik[7] using 802.11n. During the experiments, we enabled space-time block coding to exploit channel diversity in order to avoid interference.The transmission bandwidth was set to 5 MHz and provides half duplex communication.

Each UAV is remotely controlled through a dedicated GCS (Desktop and Infrastructure) as depicted in Figure 8. However, they do not take part in the routing process. Their role is to provide safety links to each UAVs for failsafe management. This is part of DGAC regulation to have at least each UAV connected to a station control via a dedicated links. Each UAV has therefore two links. The first link is either 868 MHz or 900 MHz obtained through XBEE devices[8]. The second link called safety link is a mandatory specification that must be set up to provide recovery assistance when the communication link through the Xbee is not efficient. Figure 7 illustrates the different types of data link deployed.

Test scenario and nodes mobility

During the experiment, Dr2 is firstly launched to a 800 meters distance from the GCS. When it reaches its predilection position, we launched the Dr1 to extend the mission coverage. Dr1 is used as a node forwarder (its payload is not activated) by forwarding each data packet it receives. This allowed us to send the Dr2 UAV to a 2 km distance from the GCS.

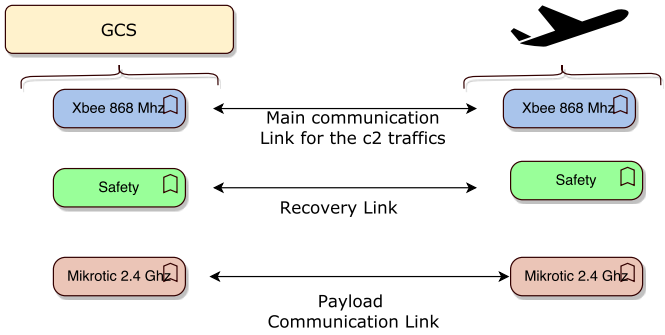

Figure 7. Communication Links Between Nodes

Furthermore, their mobility consists of rectilinear and circular movement remotely executed on-demand by the operator through the GCS desktop. The fluctuation between these mobility creates disconnection from time to time and decrease the performance accordingly.

Experimental results

Overhead

The overhead represents the amount of control packet sizes added to data packets. Within UAANETs, despite the constant mobility of nodes, the signaling (size of network control packets) cost of reactive protocols is significantly lower than payload sizes (see table VI). The overhead performance evaluation is shown in table VII

The protocol does not generate a significant amount of overhead. Thus, it does not perturb the

[7]can be seen in **http://www.mikrotik.com/**

[8]We have set 868 MHz for the Dr2 (the UAV that goes furthest) and 900 MHz for Dr1 (the nearest UAV) to avoid interference

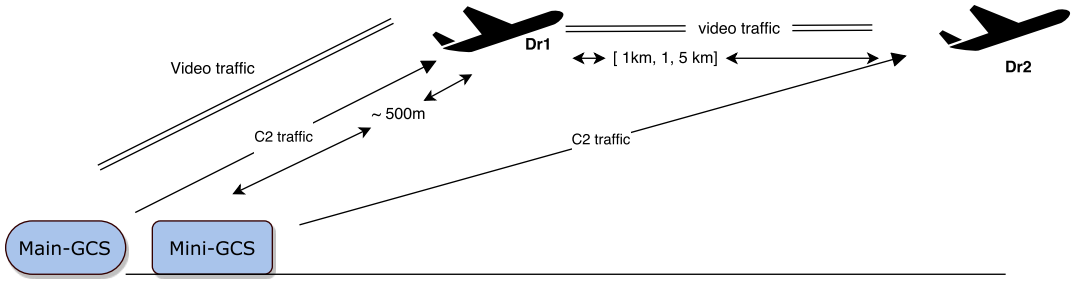

Figure 8. UAANET Real World Experiment Scenario

Table VII. Protocol Overhead over 30 minutes test

Overhead	Protocole
Control packets	352 ko
Traffic % (bytes)	2.15 %

Table VIII. Route Stability over 30 minutes Test

Metric	Value
Average route lifetime	14.328955 s
Maximum route lifetime	34.853527 s
Minimum route lifetime	4.600461 s

transfer of payload (video) packets. This good overhead results can be explained by the non-requirements to acknowledge each packet sent compared to DSR protocol and the no need to update an already established route. Indeed, once a route between GCS and Dr2 (through DR1) is computed and established, the protocol no longer seeks the complete topology of the network. Each node only checks its direct neighbor through periodic exchanged of beacon messages.

Route stability

This metric evaluates the delay during which the connectivity is uninterrupted. The result is depicted in Figure 9 and table VIII

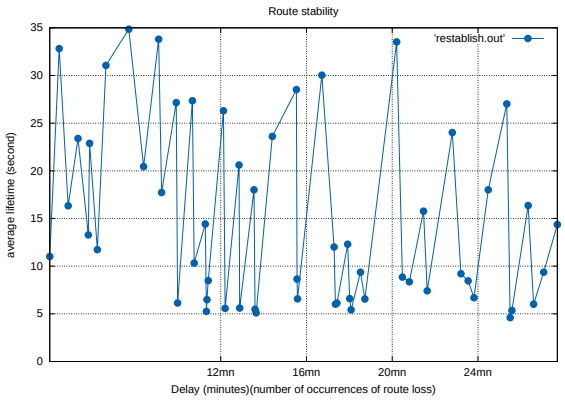

Figure 9. Route Stability Between GCS and DR2

These results show that on average over 30 minutes test, there is a route loss every 14.5 seconds. This is explained by the node mobility that creates disconnections. One important aspect that must be noted is the stability fluctuation which is caused by the difference in the radiation plane and the polarization plane of the transmitting and receiving antenna which generates a signal attenuation. Indeed, because of the 3D mobility, each UAV has its relative position slightly tilted compared to its direct neighbors position. By comparing the Wireshark trace to the mobility log of each UAV, we noticed a group maintenance packet being transmitted when UAVs mobility mode transition (For instance, from rectilinear waypoint to circular). Consequently, there is a strong likelihood that the combination of the propagation loss and the signal attenuation cause those momentary link breakages.

Moreover, apart the difference at the antenna level, it is important to mention that the introduction of real parameters in real situations impacts the performance. For example, it is noticed in the movement log that when the UAV is cornering, the inner wing mask the modem radio as it is located under the drone. Generally, nodes mobility and antenna polarization issues make it difficult to maintain communication for a long time.

Nonetheless, despite the presence of these intermittent connectivity, the recovery delay is relatively lower (order of 1 milliseconds), thus, the communi-

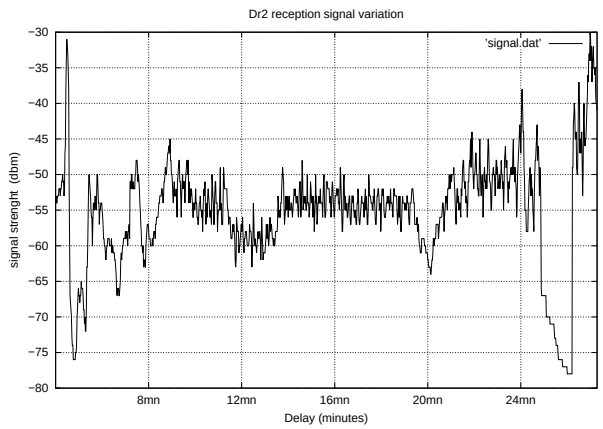

Figure 10. Variation of Reception Signal of DR2 Strength During the Mission

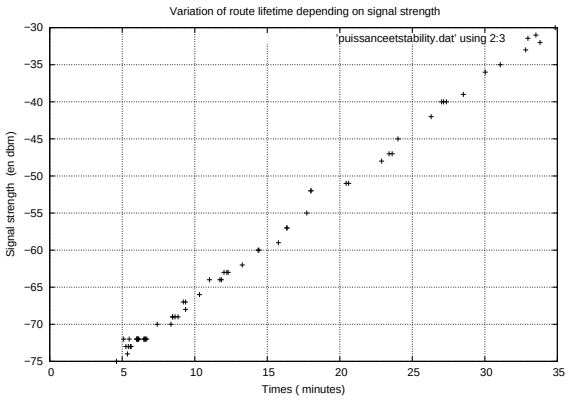

Figure 11. Average Route Stability Between DR2 and GCS

cation performance is not decreased and the payload data are correctly transmitted. As we will see in the next metric, the difference lies in the payload size being transmitted.

In the following, we will analyze the power of the Dr2 received signal to have a clear idea on route loss.

Overview of the variation of reception signal strength of Dr2 during the mission

Figure 10 shows a combination of constant and peak oscillations during the mission. It shows that the signal power fluctuates during the mission and causes link breakages when it is under -60 dbm. On the one

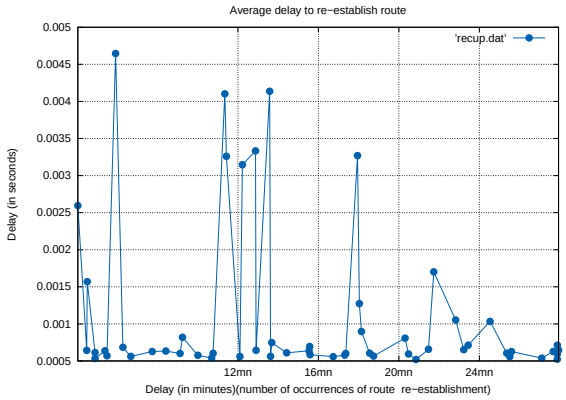

Figure 12. Average Delay for Route Establish in case of Route Loss

hand, the constant oscillation is due to the unstable movement of each UAV during its execution of c2 traffics. On the other hand, the important oscillation peak is caused both by the part of the region that presents a high attenuation and by the signal attenuation (caused by antenna polarization) as explained previously.

Additionally, based on Figure 11, which shows the correlation between signal strength and route lifetime, we can see that the route lifetime tends to increase (higher quality of service) when the signal quality increases. Since the bandwidth allocated for the data is unchanged over the whole test period, we can conclude that the noticeable fluctuation comes from other physical and topological factors.

Average delay for route establish in case of route loss

Table IX. Average Delay to Re-establish Route in case of Route Loss

Re-establish delay	Value
Average delay	0.001098 s
Maximum delay	0.004646 s
Minimum delay	0.000519 s

This metric computes the recovery time to restore from route loss. It is equal to the time interval between the sender of route request (identified with a given sequence number) and the arrival of route response to restore a route. Network connectivity

Table X. End to end delay summary

Delay	Value
Average delay	0.000153 s
Maximum delay	0.00297 s
Minimum delay	0.000100s

may be determined through the reception of broadcast hello messages. If any hello message is received from a neighbor for a given time, it indicates that neighbors are no longer within transmission range, thus, the connectivity has been lost.

Figure 12 and Table IX shows the results. We can notice that the delay is quite low given the dynamic features of UAANET. This is because of the small sizes of control packets and the speed of lights which is approximately $3 * 10^8 ms^{-1}$. Our protocol behaves correctly to its requirements specification. The topology change caused by UAVs mobility and speed does not affect significantly the delay to retrieve routes. This is also explained by the fact that the protocol does not wait for the loss of several beacon messages to start looking for an alternative route.

Average end to end delay of control packets

It refers to the time taken for a packet to be transmitted across the network. It is composed of transmission delay, propagation delay, processing delay and queuing delay.

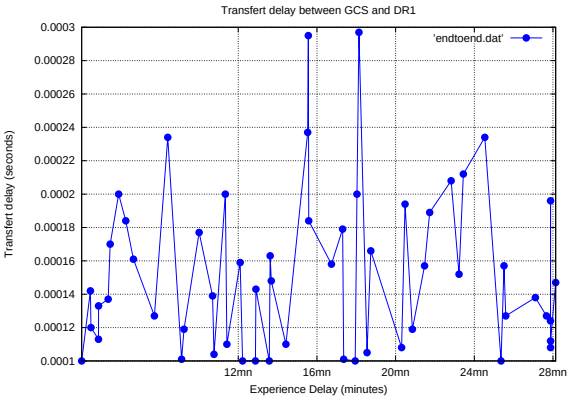

Figure 13. End to end Delay Between DR1 and GCS

Figure 14 and Table XI shows the average time routing of control packets between Dr1 and Dr2.

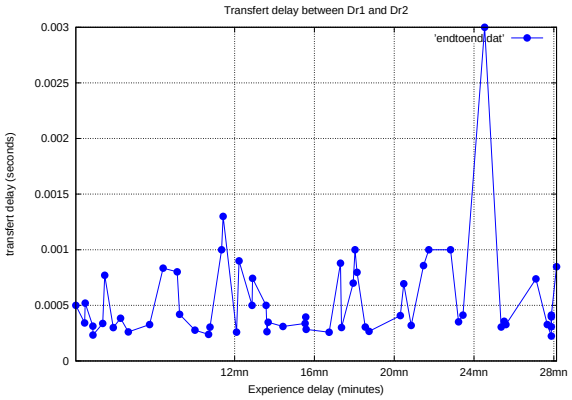

Figure 14. Average end to end Delay Between Dr1 and Dr2

Table XI. Average Delay Between Dr1 and Dr2

Delay	value
Average delay	0.000549 s
Maximum delay	0.003 s
Minimum delay	0.000224s

Figure 13 and Table X for the communication Dr1 to the GCS. These two graphs indicate a good ability to forward control traffics within the network as we are noticing small delays. The routing protocol does not add significant further delays. Besides, the small sizes of control packets justify the result. We also notice that the communication delay of Dr1 and Dr2 is slightly more important than Dr1 and GCS. This is because the GCS does not move during the duration of the mission whereas Dr1 and Dr2 execute their respective flight plan. It is important to note that the delay may vary depending on the type of control packet (request, response, hello or route error). Nonetheless, due to the small size difference between these packets, we can ignore these differences.

Loss analysis of control and payload packets

This metric measure how many routing control packets are lost during the mission. We also analyzed its impact on the payload traffics by measuring the size of payload packets being lost by the time the route is repaired. As shown in Table XII, we notice a quite important amount of control packet loss (mostly the loss of hello packets). When a certain amount of hello packet is lost, there is a delay to establish a

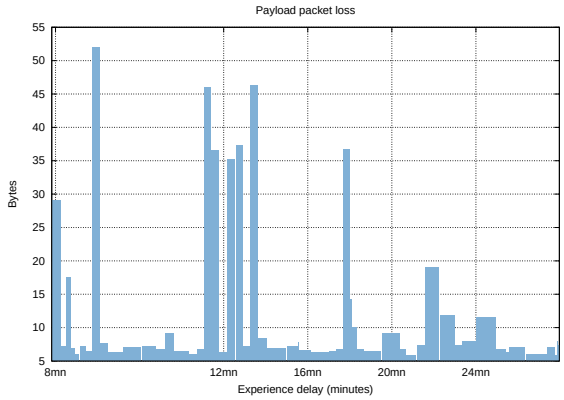

Figure 15. Average Payload Packet Loss

new route. This delay is relatively small (as depicted in Table IX) but not negligible at this level. As a result, payload packets are fragmented and decreased in size as shown in Figure 15. This indicates the degradation of the video quality viewed on the GCS desktop application during the mission.

Table XII. Control and Payload Packet Loss Summary

Parameters	Value
Number of control packet loss	284 packets
Average size of payload packet loss	12 Bytes
Maximum size of payload packet loss	52 Bytes

Conclusion and Future Research

In this paper, we have presented a case study of designing a secure UANET routing protocol applying model driven development approach. We were able to model and generate code through Mathworks (e.g., Simulink and Stateflow) tools. It allowed us an easy, rapid modeling and testing of the algorithms, and avoid many implementation problems by the use of integrated formal verification tools which gives a complete flexibility to the process. We obtained a verified and validated model that contributes to the certification of the final architecture.

We also introduced real test experiments to confront our final embedded software to the real words experiment environments. Our performance results show that our routing protocol fits well to the dynamic topology of UAANET. Despite the route loss for every 14.5 seconds which degrades the video traffic quality, we noticed that the delay to find alternative routes is relatively small, which does not perturb the payload traffic exchanges. However, we add this in our to-do list to have more stable routes and improve our communication architecture performance.

In regards to our short-term perspectives, we are currently working on formal verification of our secure routing protocol with AVISPA tool. It is a useful validation technique for authentication, non-repudiation and confidentiality. This approach is not fully applied in ad hoc networks. Some researchers had used formal analysis with a mathematical approach but it appears to be hard to automate and time consuming. Our objective is to validate the SUAP security specification. Furthermore, we also plan to perform additional real-world tests to validate our secure routing protocol with a higher node density (3 or 4 UAVs) and by considering different additional scenarios.

Acknowledgments

We would like to express our gratitude to Delair Tech research and development team for their support and help during the real world experiments.

References

[1] B. Mancini, "The great reconnaissance: a uav project in niger," *GeoInformatics*, vol. 17, no. 6, p. 6, 2014.

[2] S. Rosati, K. Kruzelecki, G. Heitz, D. Floreano, and B. Rimoldi, "Dynamic routing for flying ad hoc networks," 2014.

[3] K. P. Valavanis and G. J. Vachtsevanos, "Uav applications: introduction," in *Handbook of Unmanned Aerial Vehicles*, Springer, 2015, pp. 2639–2641.

[4] J.-A. Maxa, M. Slim Ben Mahmoud, and N. Larrieu, "Secure routing protocol design for uav ad hoc networks," in *Digital Avionics Systems Conference (DASC), 2015 IEEE/AIAA 34th*, IEEE, 2015, 4A5–1.

[5] J. Li, Y. Zhou, and L. Lamont, "Communication architectures and protocols for networking unmanned aerial vehicles," in *Globecom Workshops (GC Wkshps), 2013 IEEE*, IEEE, 2013, pp. 1415–1420.

[6] L. Gupta, R. Jain, and G. Vaszkun, "Survey of important issues in uav communication networks,"

[7] J.-A. Maxa, G. Roudiere, and N. Larrieu, "Emulation-based performance evaluation of routing protocols for uaanets," in *Communication Technologies for Vehicles*, Springer, 2015, pp. 227–240.
[8] C. Perkins, E Belding-Royer, and S. Das, "Ad hoc on demand distance vector (aodv) routing (rfc 3561)," *IETF MANET Working Group*, 2003.
[9] T. Clausen, P. Jacquet, C. Adjih, A. Laouiti, P. Minet, P. Muhlethaler, A. Qayyum, and L. Viennot, "Optimized link state routing protocol (olsr)," 2003.
[10] D. Johnson, Y Hu, and D. Maltz, "The dynamic source routing protocol (dsr) for mobile ad hoc networks for ipv4," Tech. Rep., 2007.
[11] M. Hyland, B. E. Mullins, R. O. Baldwin, and M. A. Temple, "Simulation-based performance evaluation of mobile ad hoc routing protocols in a swarm of unmanned aerial vehicles," in *Advanced Information Networking and Applications Workshops, 2007, AINAW'07. 21st International Conference on*, IEEE, vol. 2, 2007, pp. 249–256.
[12] J.-A. Maxa, "Model-driven approach to design a secure routing protocol for uav adhoc networks," in *EDSYS 2015, 15ème Congrès des doctorants*, 2015.
[13] N. Butcher, A. Stewart, and S. Biaz, "Securing the mavlink communication protocol for unmanned aircraft systems,"
[14] N. Larrieu, "How can model driven development approaches improve the certification process for uas?" In *Unmanned Aircraft Systems (ICUAS), 2014 International Conference on*, IEEE, 2014, pp. 253–260.
[15] *Delair Tech*, www . delair - tech . com, [Online; accessed 09-octobre-2015], 2015.
[16] Y. Moy, E. Ledinot, H. Delseny, V. Wiels, and B. Monate, "Testing or formal verification: do-178c alternatives and industrial experience," *Software, IEEE*, vol. 30, no. 3, pp. 50–57, 2013.
[17] N. Larrieu and A. Varet, "Prototypage rapide de logiciel pour les systèmes avioniques: contribution des approches orientées modèle pour la certification de systèmes complexes," 2014.
[18] R. Castanet and D. Rouillard, "Generate certified test cases by combining theorem proving and reachability analysis," in *Testing of Communicating Systems XIV*, Springer, 2002, pp. 249–265.
[19] B. Kannhavong, H. Nakayama, Y. Nemoto, N. Kato, and A. Jamalipour, "A survey of routing attacks in mobile ad hoc networks," *Wireless communications, IEEE*, vol. 14, no. 5, pp. 85–91, 2007.
[20] M. Khabbazian, H. Mercier, and V. K. Bhargava, "Severity analysis and countermeasure for the wormhole attack in wireless ad hoc networks," *Wireless Communications, IEEE Transactions on*, vol. 8, no. 2, pp. 736–745, 2009.
[21] J. Cordasco and S. Wetzel, "An attacker model for manet routing security," in *Proceedings of the second ACM conference on Wireless network security*, ACM, 2009, pp. 87–94.
[22] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Packet leashes: a defense against wormhole attacks in wireless networks," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, IEEE, vol. 3, 2003, pp. 1976–1986.
[23] M. Jain and H. Kandwal, "A survey on complex wormhole attack in wireless ad hoc networks," in *2009 International Conference on Advances in Computing, Control, and Telecommunication Technologies*, IEEE, 2009, pp. 555–558.
[24] O. Team, *Openembedded user manual*, 2006.
[25] I. D. Chakeres and E. M. Belding-Royer, "Aodv implementation design and performance evaluation," *International Journal of Wireless and Mobile Computing*, vol. 2, no. 3, p. 42, 2005.